1968

A disc-oriented fortran iv processor for the GE225 computer

Howard E. Clark
Lehigh University

A DISC-ORIENTED FORTRAN IV PROCESSOR

FOR THE GE225 COMPUTER

by

Howard Evans Clark

A Thesis

Presented to the Graduate Committee

of Lehigh University

in candidacy for the Degree of

Master of Science

in the

Information Sciences

Lehigh University

1968

This thesis is accepted and approved in partial fulfillment of the requirements for the degree of Master of Science.

May 17, 1968
(date)

Daniel J. Hillman
Professor in charge

Daniel J. Hillman
Head of the Department

Table of Contents

Abstract

A disc-oriented processor for programs written in the FORTRAN IV language has been developed for the General Electric Model 225 computer at Lehigh University. The general characteristics of the processor are:

(i) it compiles FORTRAN IV programs and loads them into memory for execution;

(ii) it stores users' program overlays on disc rather than tape, so that they do not have to be read sequentially;

(iii) it allows users to store intermediate results on disc during the execution of their programs;

(iv) it allows users to store their programs in a disc program library and to recall them with a single control card; and

(v) it provides the means for purging entries in the disc program library.

The processor is an outgrowth of a tape-oriented system distributed by General Electric. The disc system has been shown to compile and execute FORTRAN IV programs in as little as 23% of the time required by the tape system.

I. Project Goals

Lehigh University's current computing system consists of a General Electric Model 225 computer and a number of input and output devices. The major components of the system configuration are listed in Appendix A in order to indicate the minimum configuration for use of the FORTRAN IV processor; at this point it need only be mentioned that the central memory consists of 8192 words and that a disc file and four magnetic tape units are included in the configuration.

The computer language that is most commonly used at Lehigh is FORTRAN IV. The FORTRAN IV compiler that has been furnished by the General Electric Computer Department for its 200 series machines is designed to use only magnetic tapes for auxiliary storage; it cannot take advantage of an available disc file. There are of course inherent advantages of random access storage for many applications, so it was desirable to investigate the feasibility of creating a disc-oriented FORTRAN IV system for use at Lehigh.

Specific project objectives included the storage of the entire FORTRAN IV processor on disc, the storage of users' program overlays on disc and the provision of a means to enable users to store their programs and data on disc.

The disc-oriented system does now exist; a description of its characteristics and of the problems encountered in its development is given in the succeeding sections.

The development of the disc system was based on modifications of and substantial additions to General Electric's tape system (Program Number CD225H6.004; see Reference 1). The programming effort and computer time that would have been necessary for a completely new system would have made the project infeasible.

II. Characteristics of the GE225 Tape and Disc Units

The tape units in Lehigh's 225 configuration are Model 680's, which have a transfer rate of 15,000 characters/second (see Reference 2 for a complete description). Tapes are written at a density of 200 characters/inch; the tape moves at 75 inches/second, so the transfer rate is 15 kc. Four six-bit characters are needed to represent a 20-bit binary word, so the transfer rate for tapes written in binary mode is 3750 words/second. An inter-record gap is .75 inches long; stop/start time between records is .018 seconds. Tapes are rewound at 150 inches/second.

General Electric also offers a Model 690 tape unit with a transfer rate of 41.6 kc, which compares much more favorably with the transfer rate of the disc file than does the rate of the Model 680.

The GE-200 Series Disc Storage Unit can contain a maximum of 16 discs, each having a storage capacity of 393,216 words (see Reference 3 for full details). The smallest amount of information that can be read or written is a frame, which consists of 64 words. The average access time for a frame is .225 seconds. Two-thirds of the data stored on a disc can be transferred to or from memory at a rate of 23,810 words/second; the remainder can be transferred at 11,905

words/second. The average transfer rate for the disc file is thus $\frac{2}{3}(23,810)+\frac{1}{3}(11,905) = 19,842$ words/second.

III. Major Components of the FORTRAN IV Processor

A brief discussion of the structure of the FORTRAN IV processor is necessary. The principal components of the system are:

1) The monitor, which interprets control cards and calls the appropriate segment of the processor.
2) The compiler, which translates statements in the FORTRAN IV language into machine language instructions.
3) The relocatable subprogram library, which is a collection of frequently used subprograms that are combined with users' programs when required. The subprograms are called "relocatable" because they can be loaded into any part of memory (the loader modifies them so as to make them "absolute" when they are loaded).
4) The absolute suffix, which is a collection of subprograms that perform input-output and arithmetic functions. The subprograms are called "absolute" because they are loaded into a fixed portion of memory and they form a "suffix" because they are the last portion loaded before a user's program is executed.

5) The loader, which stores the program generated from FORTRAN statements into memory, loads the library subprograms that have been referenced, reads in the absolute suffix and initiates execution.

IV. Description of the Disc FORTRAN IV Processor

A. Searching the Relocatable Library

The relocatable library consists of 63 subprograms which perform frequently required computation, such as the evaluation of trigonometric functions, the conversion of floating-point numbers to integers, etc. A complete list of the names of the subprograms may be found in Appendix B. Some library subprograms have more than one name (each name corresponds to an entry point); there are 111 entry points for the 63 subprograms.

In the tape FORTRAN processor, each subprogram is stored on tape as a sequence of binary card images. The first card image is a "header" which declares the name(s) of the subprogram, the entry locations and the names of other routines that are referenced in the subprogram. On tape, it is evident that each subprogram must precede the other subprograms that it references, since they are read sequentially. The loader reads through the entire library (210 card images) every time a user's program is loaded. The loader must read through the whole library because the absolute suffix is stored on tape after the library routines;

it cannot be read earlier because it must occupy the same memory locations as the loader.

Using disc storage, there are two ways that the library routines can be organized. They can be read sequentially, with the loader picking out the ones that are referenced, or they can be read individually from random areas of disc storage as the loader discovers that they are needed. The latter approach seems most natural and efficient, but it proved impractical in the case of Lehigh's computer. To read only the required subprograms, the loader would have to maintain tables that listed all the names of library routines, the names of routines that they reference, their locations on disc and their lengths. The memory space that would be required for such tables and the necessary buffers for reading the routines proved to be prohibitive. Obviously the loader must coexist in memory with the user's main program and required library routines; the 8192 memory locations of Lehigh's 225 dictate that a fairly simple loader be used. In addition, it is doubtful that it could be shown that a time saving would result from random access to library routines. While the average subprogram would require only two

frames of disc storage and thus the average transfer time would be about .006 seconds, the average access time is .225 seconds. Thus very few subprograms could be read in the time that it takes to read the entire library sequentially (using reasonably sized buffers).

Thus in the disc processor, the library routines are read sequentially as if from tape (except that buffers are much larger than in the tape processor). There are two significant advantages over tape storage, though. One is that the transfer rate is considerably greater and the other is that reading terminates when there are no unsatisfied references. The library routines have been reordered according to the estimated probability of their being required by a user's program; of course, the previously mentioned constraint that no routine may reference a preceding routine still applies. The reading of the library routines is begun before the user's main program is loaded; since the subprogram "EXIT" is used by virtually all programs, it is in core and ready to be added when the loading of the main program is complete.

Appendix B lists the names of the library routines in the order in which they are read from disc.

B. Treatment of Overlays

A user's program cannot be longer than 5112 locations because of the length of the absolute suffix. This rather severe restriction necessitates the segmentation of many programs; a main program may need to call a number of overlays during the course of its computation. For example, a program might need to call three subroutines during execution, but none of the three calls another. In this case, the three subroutines could share the same memory locations; each of them could be stored on tape or disc and read into memory when needed.

The tape processor writes a maximum of twenty overlays, as defined by the programmer, on tape and reads them when the library subroutine "OVRLAY" is called (the user gives the number of the desired overlay as an argument). The subroutine OVRLAY must keep a record of the last overlay read to know how many tape records must be bypassed to reach the desired overlay. The tape units are capable of reading records backwards, so the situation is not as bad as it might be. The tape system actually writes two records on tape for each overlay, though; the first is a four-word record that specifies where

in memory the overlay is to be loaded and the second is the overlay itself.

The advantage of random access to records is quite evident in the use of overlays. If a program uses several overlays in a sequential order, tape storage is not inefficient. But if the program must call overlays in a random manner, then a rather heavy penalty in computer time accompanies the use of tape. Section V discusses the difference in efficiency between the tape and disc processors in handling overlays.

The disc processor stores overlays in a reserved area on disc and forms a table containing the locations and lengths for the subroutine OVRLAY. The table is contained in memory during execution, so OVRLAY does not need to read two records to obtain a single overlay, as in the tape system. Also, the disc system allows a maximum of thirty overlays; twenty was found to be restrictive to some users.

One rather unfortunate discrepancy between the tape and disc processors resulted from the loader's handling of disc overlays. The design of the disc file requires that the number of frames (1 frame = 64 words) transferred to or from memory must be

an integer and that the first location used in a transfer must be a multiple of 64. Thus any disc input or output must begin and end at a 64-word boundary. The only really practical way to handle disc overlays was to have the loader increase the current loading location to the nearest multiple of 64 when it finds that an overlay is to be loaded. This procedure is slightly wasteful of memory space and introduces an incompatibility between the tape and disc versions. However, it is assumed that the tape system would be used only as "backup" for the disc system, in which case the situation is not too serious. Any overlays loaded by the disc system will fit into memory when loaded by the tape version.

One of the project goals was to provide the means for programmers to use the disc file for intermediate storage during program execution. The OVRLAY subroutine offered a natural way to accomplish what was wanted. By adding a few instructions and creating a new entry to the subroutine, called WRTOVR, users were given the capability of writing specified overlays on disc as well as reading them. Thus a programmer can perform operations on a set of data, write it on disc, use the same area of memory for

another purpose, and recall the data from disc when necessary. To use WRTOVR, the programmer specifies the overlay number as an argument, just as for OVRLAY.

C. Permanent Storage of Programs

As previously mentioned, it was hoped at the outset that users could be given the means to store their programs in a permanent disc library area and to recall them at will. The ability to store programs in this manner can save a great deal of computer time if a program is executed frequently. A long program, consisting of several overlays, may take perhaps two minutes to be loaded from a deck of binary cards. Under the tape system it is possible to reuse the tape that contains a program in the form of binary card images, but doing so entails not only the mounting of the tape but the subsequent writing of overlays on another tape by the loader. The process is generally not any quicker than loading the program through the card reader unless the tape can be mounted well in advance of its use and/or jobs requiring the program can be grouped together. In any event, it is readily seen that a disc library

of heavily used programs is highly desirable. Any program in such a library can be loaded and execution begun in approximately one second.

The necessary software for creating and maintaining a program library has been incorporated into the disc system. The maximum number of programs that the library can contain is presently 128, but the number is easily changed. The total amount of disc space available for the library is stored as a parameter in "PROGRAM SAVER", the routine that enters a program into the library. To accommodate 128 programs of maximum length (but using no overlays), a total of 672,256 words of disc storage, or 1.72 discs, would be required. The total includes 512 words for the directory. Two discs would seem to be a reasonable allocation for a library of programs that might require overlays.

To enter a program into the library, a user need only include in his deck a control card having "$SAVE" in columns 1-5 and a name of up to six characters, starting in column 7. The $SAVE card must appear before the $LOAD or $DATA control card that initiates loading. When the loader has finished

loading the program, it will read PROGRAM SAVER, which searches the directory. If the name that is specified in columns 7-12 is found, the program cannot be entered into the library and so an error message is printed. If the name does not appear in the directory, PROGRAM SAVER determines whether there is enough remaining space in the library area to store the program. If not, it prints a message and exits. If there is sufficient space, it stores the program and enters the name, location and length into the directory. If the program includes overlays, they are copied from the original storage area into the library area. PROGRAM SAVER then exits to the program for execution.

Library programs are thus stored in absolute form, ready for immediate loading. Of course it would be possible to store them in relocatable form and to allow users to combine individual subprograms in the same way that they are loaded from the FORTRAN system relocatable library, but doing so would conflict with the real intent of the library. Library programs are intended to be ones that are used frequently but changed infrequently.

Storing them in relocatable form would increase loading time as well as the complexity of the loader.

To rerun a program that is stored in the library, a user needs only a single control card containing "$RUN" in columns 1-4 and the name of the program in columns 7-12. The FORTRAN monitor will recognize "RUN" and read in a program called "DIRECTORY SEARCH" which, appropriately enough, looks for the name in the directory. If the name is not found, a message is printed and the run is terminated. If the name is found, the corresponding program is read into memory. The absolute suffix is then read and execution of the program is begun. Any necessary data cards are supplied by the user, following the "$RUN" card.

One restriction placed on users of library programs is that a program cannot call WRTOVR to write an overlay if the program has been loaded from the library. An attempt to do so results in an error exit. In this way the library is protected from erroneous overwriting. The library can be otherwise protected from destruction

by setting switches that are located on the disc file unit. Each disc has a corresponding switch that, when thrown, allows the disc to be read but not written. Thus whenever a program is entered into the library, the computer operator may be required to physically enable the writing on the affected discs.

A basic attitude underlying the design of the disc library software is that Computing Center staff members must strictly control the addition and deletion of library programs. If disc space were not too limited, it would be desirable to allow users to create as many library entries as they felt necessary. However, the deletion of programs cannot be left to the discretion of any user, particularly in a university environment. For this reason, it is assumed that a staff member will maintain a list of authorized library programs and initiate the deletion of unauthorized entries when disc space is at a premium. To accomplish this, a program called "MAINTAINER" has been developed. MAINTAINER reads data cards and carries out their commands until a card containing "END" in columns 1-3 is encountered. A data card containing "LIST"

in columns 1-4 will cause a list of the directory to be printed. A data card containing "DELETE" in columns 1-6 will cause the program whose name appears in columns 7-12 to be deleted from the disc library, if the program name appears in the directory.

V. Efficiency of the Disc Processor

The disc FORTRAN system has been shown in practice to compile and load programs in approximately 50% of the time required by the tape system. The compiler in the disc version has not been changed so as to generate machine language instructions that differ from those generated by the tape compiler, but the compiler has been speeded up considerably. One modification that was made involved the use of the MOV instruction in many parts of the compiler; the MOV instruction relocates large blocks of data in memory in about 30% of the time required by a program loop which loads and stores registers (see Reference 4). Since the MOV instruction is an optional feature of the GE225, the programmers of the tape compiler could not assume its availability.

Much of the increased efficiency of the disc system derives from the random access capability and high information transfer rate of the disc file. To illustrate the advantages that the disc file offers, consider a FORTRAN program that consists only of the statements "CALL EXIT" and "END". To process the program, the following sequence of events must occur. First a load card is read and

executed; it reads the FORTRAN monitor into memory. The monitor interprets the control cards and learns that a program is to be compiled. It then reads a section of the compiler called COMPILER A (the whole compiler cannot fit into memory at once). COMPILER A's purpose is to process specification statements, such as "DIMENSION", "COMMON", "COMPLEX", etc. It finds none, so it reads in the second section of the compiler, COMPILER B. COMPILER B compiles the program, writing the binary card images on tape and punching them if necessary. COMPILER B sees the "END" statement, so it transfers back to the monitor to process more control cards. The monitor reads "$LOAD", which causes it to read the loader and transfer to it. The loader then loads the program from tape, reads through the relocatable library and loads EXIT. The absolute suffix is read and execution of the program is begun. EXIT is called; it rereads the monitor, which processes more control cards.

It is of interest to calculate how long the tape and disc processors would take to perform the above operations. The disc processor needs to read just one record (384 words on disc) of the relocatable

library to find EXIT, while the tape processor must read the entire library (210 40-word records). The tape processor must wait for the system tape to rewind after the absolute suffix has been read; this may be done concurrently with execution of a program that is considerably longer than "CALL EXIT". The case is an optimal one for the disc processor, but it does give an idea of the difference in performance that can be expected. The time required to read and process the load card, control and program cards is not included in the calculations; it will be less for the disc system than for the tape system. The lengths of the records that must be read are given below for tape and disc (recall that a disc record length must be a multiple of 64).

		Tape	Disc
MONITOR		1124	1152
COMPILER A		5382	5440
COMPILER B		3777	3840
LOADER		1543	1600
LIBRARY	210x40=	8400	384
ABSOLUTE SUFFIX		2735	2752
MONITOR		1124	1152
		24085	16320

If an average access time of .15 seconds is assumed for the disc records (which can be attained with efficient record placement), the total access time for the seven records is 7x.15 = 1.05 seconds. The average transfer rate is 19,842 words/second, so the total time required is $1.05 + \frac{16320}{19842} = 2.87$ seconds.

The tape processor must read a total of 216 records from the system tape. Stop/start time between records is .018 seconds, so the total stop/start time is 215x.018 = 3.87 seconds. The Model 680 tape units can transfer 3750 words/second, so reading the tape takes $3.87 + \frac{24085}{3750} = 10.29$ seconds. Rewinding the tape takes 4.14 seconds, so the total time for this case is at least 14.43 seconds. Thus the disc processor takes $\frac{2.87x100}{14.43} = 20\%$ of the time required by the tape system (using Model 680 units) if card reading and compiling time is not included.

It is also of interest to perform the same calculations for the Model 690 tape units to see what advantage the disc system might offer. The results are:

Stop/start time	3.87 seconds
Transfer time	2.32 "
Rewind time	2.17 "
Total	8.36 "

Thus in this example, the disc system would take $\frac{2.87 \times 100}{8.36}$ = 34% of the time required by the tape system using Model 690 units.

The above example was actually run under the disc and tape processors at Lehigh. The disc system took 5.8 seconds to compile and execute the program; the tape system took 25 seconds. Based on the above calculations, the 5.8 seconds required by the disc processor is roughly what would be expected; about three seconds were needed for reading and interpreting control cards and compiling the program. The 25 seconds taken by the tape system seems excessive in light of the calculations and the time required by the disc system; the difference is a result of the more efficient coding of the disc version. For this case, the disc version needed only 23% of the time taken by the tape version; for longer programs, especially those which use more library routines, the ratio would not be

as small.

A rather large difference in execution times of programs that use overlays can be expected if they are run under both the tape and disc systems. As previously mentioned, the tape processor not only must read overlays sequentially, but actually reads two records for each overlay. To learn how much faster the disc version can read overlays than the tape version, a test program was constructed. The program called 20 overlays (each 4000 words in length) in sequence. The disc system loaded the program from binary cards and executed it in 46 seconds; the tape system took 100 seconds. Thus the disc system can be expected to read a number of overlays in much less than 46% of the time taken by the tape system, since overlays are not always used in the order in which they are initially loaded.

VI. Availability of Programs

The listings of programs that comprise the disc processor, as well as test runs, are far too voluminous to be included here. They will be retained in the author's files until June, 1973. The programs will require minor modifications to be used at installations other than Lehigh's, because they were designed to be compatible with Lehigh's monitor system.

VII. Conclusions

The design and programming of a disc-oriented FORTRAN IV processor for the GE225 has been completed. The system has been implemented at Lehigh University and tested rather extensively. In comparison with the General Electric tape FORTRAN IV processor, the disc processor offers substantial savings in computer time and increased capabilities. An installation that must process a high volume of FORTRAN programs might justify the expense of a disc file on the basis of the increased throughput that can be achieved with the use of the disc processor.

Appendix A

Configuration of Lehigh University's GE225 System

The components that are required for use of the disc FORTRAN IV processor are starred or otherwise noted:

1 - Central processor with 8,192 word memory*
4 - Model 680 magnetic tape units (one tape is used by the disc processor for loading programs)
1 - Disc file unit, with 12 discs (2 discs are used for storage of the FORTRAN processor and the program library)
1 - 400 card/minute reader*
1 - 900 line/minute printer*
1 - 100 card/minute punch (required for output of program binary deck)
1 - Auxiliary arithmetic unit*
1 - Console typewriter
MOV instruction*
BCD arithmetic
Automatic priority interrupt

Appendix B

Contents of the FORTRAN System Relocatable Library

The system relocatable library is read from disc in a different order than the one in which it appears on the tape released by General Electric. The revised order is given below (see Reference 1 for an explanation of the purpose of individual routines). The entry names are given for the 63 routines, as well as the numbers of the other routines referenced.

Number	Entry Names	Referenced Routines
1	EXIT	
2	,RI/,WO	
3	,TR/,TW	
4	,BS/,RW/,EF	
5	,A1/,A2/,A3/,A4	
6	,I1	
7	,I2	
8	,I3	29,35
9	IABS/ABS	
10	IDIM	
11	DIM	
12	MIN1/AMIN1/MAX1/AMAX1	
13	MINO/AMINO/MAXO/AMAXO	
14	FLOAT	
15	FIX	
16	AINT/INT	
17	ISIGN	
18	SIGN	
19	MOD	
20	AMOD	

Number	Entry Names	Referenced Routines
21	,L1/,L2	
22	,R1/,R2/,R3/,R4/,R5/,R6	
23	OVRLAY/WRTOVR	
24	DSQRT	28,54,56
25	CSQRT	27,28
26	CLOG	27,29,30
27	CABS	28
28	SQRT	
29	ALOG10/ALOG	
30	ATAN/ATAN2	
31	CEXP	33,35
32	CCOS/CSIN	33,35
33	COS/SIN	
34	TANH	35
35	EXP	
36	DBLE	
37	IDINT	
38	SNGL	
39	DABS	
40	DMOD	54,56
41	DSIGN	
42	DCOS/DSIN	54,56
43	,I4/,I5/,I6/,I7	44,45,56
44	DEXP	54,56
45	DLOG/DLOG10	54,56
46	DATAN/DATAN2	54,56
47	DMIN1/DMAX1	
48	,I8	52,53
49	CONJG	
50	CMPLX	
51	REAL/AIMAG	

Number	Entry Names	Referenced Routines
52	,C1/,C2/,C3/,C4/,C5/,C6	
53	,C7/,C8	
54	,D8	56
55	,D7	56
56	,D1/,D2/,D3/,D4/,D5/,D6	
57	PDUMP/DUMP	
58	,ST	
59	OVERFL/DVCHK	
60	XIFEOF	
61	SSWTCH	
62	,PA	
63	SLITE/SLITET	

References

1. General Electric Company, GE200 Series FORTRAN IV Reference Manual, XCPB-1324, Phoenix, Arizona, 1967.

2. General Electric Company, GE200 Series Magnetic Tape Subsystem, CPB-339B, Phoenix, Arizona, 1966.

3. General Electric Company, GE200 Series Disc Storage Subsystem, CPB-323B, Phoenix, Arizona, 1966.

4. General Electric Company, GE225 Programming Reference Manual, CPB-252A, Phoenix, Arizona, 1964.

Vita

Howard Evans Clark was born in Stuart, Virginia, on May 17, 1940, the son of Jewell T. and William H. Clark. In June, 1958, he was graduated from Douglas Freeman High School in Richmond, Virginia. He received a B.S. in Mathematics in June, 1962, from Virginia Polytechnic Institute, where he was a member of the chapters of Pi Mu Epsilon and Phi Kappa Phi. He is married to the former Mary Carol Rose of Bethesda, Maryland, and they have one child, Catherine Anne.

Since his graduation from Virginia Polytechnic Institute, Mr. Clark has been employed by the National Aeronautics and Space Administration, the United States Army and Lehigh University.